REMARKS

The Office Action dated October 5, 2005, has been reviewed in detail along with all references made of record. Reconsideration of the claims of the instant application is respectfully requested in view of the following remarks.

Applicant would like to extend its appreciation to the Examiner for the time and attention accorded this case. As will be set forth in detail here below, the issues raised by the Office in the outstanding Office Action, when reconsidered in light of the foregoing amendments and the following comments, should be resolved in Applicant's favor.

As of the mailing date of the outstanding Office Action, Claims 1-28 were pending in the instant application. By this Amendment, Claims 1, 3, and 21 are amended.

Claims 1-28 stand rejected under 35 U.S.C. 102(b) as "being anticipated by Bierbaum et al...". In view of the comments provided herebelow, it is respectfully submitted that Claims 1-28 fully distinguish over the applied art. By this Amendment, several amendments have been made to help further clarify the uniqueness of the claims over the applied art. Reconsideration and withdrawal of the present rejection are thus hereby respectfully requested.

Independent Claims 1 and 21 involve generating machine control instructions through a process that includes connecting selected objects to a network of objects, generating machine control instructions from objects in the network of objects, and effecting updates within the aggregate of the network of objects and the connections between connected objects to accurately reflect any changes made to the machine control instructions generated in said generating step.

The Examiner cites Bierbaum as anticipating Claims 1 and 21. However, there appears to be some confusion with regard to the generating of machine control instructions and the effecting of updates to reflect changes made to **generated** machine control instructions. The machine control instructions used in the dVISE system (Bierbaum pg. 3-23 to 3-26) consist of commands in a scripting language embedded in objects. However, these machine control instructions are not generated from a network of objects, but are written by humans. They are interpreted at runtime by the dVS runtime environment, in contrast to Claims 1 and 21, which involve an explicit step of generating machine control instructions. dVISE operates directly on these scripting commands rather than generating any additional machine control instructions (pg. 3-24, 2nd and 3rd paragraphs with boxed software code).

Unlike dVISE, the CAVE Library actually does generate machine control instructions. There are two types of machine control instructions used in the CAVE Library (Bierbaum pg. 3-20 to 3-22), C/C++ source code instructions and binary machine code instructions. Only the binary machine code qualifies as "generated machine control instructions", since the C/C++ source code instructions are used by the compiler to generate the binary machine code machine control instructions. However, the CAVE Library does not provide any way to effect updates to accurately reflect any changes made to the machine control instructions generated (Claim 1) or any way to read in a sequence of generated machine control instructions (Claim 21), since it has no way to understand or read binary machine code instructions generated by a compiler back into the system, only the ability to generate such instructions initially. Bierbaum (pg. 3-20, Description, 3rd paragraph) only teaches "updating program data" via callbacks executed at runtime. While this can be used to update the attributes of virtual objects in the CAVE system, these updates have nothing to do with the generated machine control instructions, which consist

solely of binary machine code in the CAVE system. Any changes made to the generated compiled binary machine code instructions are not reflected in the virtual objects in the CAVE system at all -- in fact, these virtual 3-D objects have no direct correspondence with the generated binary machine code output by the compiler.

One might speculate that it might be possible to combine the teachings in Bierbaum of the dVISE system (pg. 3-23 to 3-26) and the CAVE Library (pg. 3-20 to 3-22) to arrive at the teachings covered by Claims 1 and 21. While it may be possible to combine the two software packages described to create a system for visualizing interacting with simulated virtual 3-D objects in a virtual environment, that is not what is being claimed in the above claims. Their teachings cannot be readily combined to arrive at Claims 1 and 21. Even if one was somehow able to extend dVISE to generate machine control instructions from these scripting commands (for example, by writing a compiler that turned the scripting commands into compiled code, which is **not** anticipated by Bierbaum), there would be no way to affect the original objects (the ones containing the scripting commands) by changing the **generated** machine control instructions, as claimed in Claims 1 and 21. Modifying any **generated** machine control instructions would only affect the generated code, not the original objects containing the scripting commands, nor would it automatically change which objects are connected. The "generated events and data updates" cited in Bierbaum (pg. 3-25, High-level Scripting Interface) are **not** a means for accomplishing this, since the system lacks any means for identifying specific changes to the **generated** machine control instructions and mapping those changes back to the original aggregate of the network of objects and the connections between connected objects, as taught in the present invention. For example, if the user modifies an ADD instruction generated by a compiler from the original objects, the system has no means of updating the network of

objects to reflect this change, since it cannot isolate this instruction and map it to the appropriate place in the scripting commands of the original object it was generated from, which is what would be required to accurately reflect changes made to the generated machine control instructions. There is simply no way for these systems described in Bierbaum to identify which part of an object the modified ADD instruction came from, and so no way to update the appropriate portions of scripting commands in the original objects to accurately reflect this change, in contrast to the teachings of the present invention.

To further clarify this distinction, Claim 1 has been amended to specify that the generation of machine control instructions specifically involves producing a new sequence of machine control instructions from multiple objects in the network of objects, in contrast to the dVISE system cited in Bierbaum, where a given sequence of machine control instructions (in addition to the fact that they are not generated) corresponds to a single object. Despite the fact that some objects may affect each other, each object has its own series of scripting commands (Bierbaum, pg. 3-24, 2nd paragraph with box software code), so editing the scripting commands in one object will not modify **the actual scripting commands** in another. Similarly, Claim 21 has been amended to clarify that multiple objects and their connections in a network of objects are built from a single sequence of generated machine control instructions, in contrast to Bierbaum, where no method is taught for extracting a high-level model from compiled code and which does not provide a mechanism for identifying which instructions and connections in which objects were modified in an aggregate sequence of instructions made from multiple objects. Claim 21 has also been amended to clarify that the machine control instructions referred to are computer-generated, in contrast to the scripting commands written by humans in the dVISE system cited in Bierbaum.

In addition, the Examiner cites Bierbaum (pg. 3-20, Description, 2nd paragraph) as anticipating the portions of Claims 1 and 21 involving a network of objects. The cited passage in Bierbaum that refers to a network is the sentence "a running CAVE Library application is composed of several processes for handling devices, networking, and so on." The reference to networking in the sentence clearly refers to the function calls necessary for communicating with other computers on a computer network, not to any conceptual network of objects in memory or storage containing sets of machine control instructions. The network referenced in Bierbaum consists of computers, **not** objects in an object library that contain sets of machine control instructions. While the phrase "network of objects" does indeed contain the word "network", it has little in common with the computer network implied by the term "networking" in the passage cited above, other than being a topological structure of connected nodes. In fact, Bierbaum pg. 3-20, Description, 1st paragraph clearly states that the CAVE Library, "does not include higher-level features like collision detection or built-in support for object behaviors." The CAVE Library only handles access to the input and output of the connected computer network (Bierbaum pg. 3-20, Description, 1st paragraph, 2nd sentence), it does not address any form of "network of objects" as Claims 1 and 21 do. Indeed, Bierbaum pg. 3-21, 1st paragraph explains the use of networking in the CAVE Library, "The CAVE Library automatically transmits user and tracker information between all connected CAVEs, but the application is responsible for transmitting whatever other information must be shared (a function exists to transmit data to the remote applications)." This sentence shows that the use of networking is to transmit user, tracker, and application data over a network, rather than using a network of objects to generate machine control instructions.

Claim 3 has been amended to clarify that it refers specifically to executing functions contained in the objects in the network to generate machine control instructions, rather than copying them, compiling them, or some other method of derivation, which are alternative embodiments of the present invention covered by Claim 1.

In regards to Claim 4, the Examiner cites Bierbaum (pg. 3-24, 2nd paragraph with boxed software code) as teaching of computer instructions to load the code libraries represented by the objects. However, the computer instructions in the cited reference are high-level scripting instructions to control the behavior of virtual objects, and these scripting instructions do not load any code libraries, they are just interpreted to control how 3-D objects on a display behave. Indeed, they cannot "load the code libraries represented by the objects" and stated in Claim 4, since the objects referred to in the cited passage of Bierbaum (pg. 3-24, 2nd paragraph with boxed software code) do not represent code libraries at all -- they represent virtual 3-D objects.

In regards to Claim 6, the Examiner cites Bierbaum (pg. 3-24, 2nd paragraph with boxed software code) as teaching of an object of machine control instructions. However, Claim 6 specifically refers to machine control instructions being "computer instructions belonging to an instruction set architecture." The so-called "machine control instructions" cited in Bierbaum are clearly lines of a scripting language, not computer instructions belonging to an instruction set architecture. As an interpreted language, the commands in the scripting language referenced in Bierbaum (pg. 3-24, 2nd paragraph with box software code) cannot be considered to belong to an instruction set architecture by even the broadest interpretation, as they are high-level scripting

commands, not low-level instructions. Bierbaum (pg. 3-23, Description, 2nd paragraph) even explicitly states that it uses, "very high-level interfaces."

In view of the foregoing, it is respectfully submitted that Claims 1 and 21 fully distinguish over the applied art. By virtue of dependence from Claims 1 and 21, it is respectfully submitted that Claims 2-20 and 22-28 similarly distinguish over the applied art. As mentioned above in connection with the 102 rejection, several claim amendments have been made to provide even greater clarity in more fully distinguishing various features of the present invention over the applied art.

In view of the foregoing, it is respectfully submitted that Claims 1-28 are in condition for allowance. Accordingly, reconsideration and withdrawal of the present rejection are hereby respectfully requested.

References Made of Record but not Applied:

The references made of record and considered by the Office to be "pertinent to applicant's disclosure" have been reviewed. Applicant acknowledges that the Office has deemed such references not sufficiently relevant to have been relied upon in the outstanding Office Action. However, to the extent that the Office may apply such references against the claims in the future, Applicant is prepared to fully respond thereto.

* * *

In summary, Applicant respectfully submits that the instant application, including Claims 1-28, is presently in condition for allowance. Notice to the effect is hereby earnestly solicited.

Respectfully submitted,

Dated: February 1, 2006

Rusty S. Lee

1525 Wilder Ave. #606
Honolulu, HI 96822

Applicant